Aprendizaje Automático y Minería de Datos

Redes Neuronales

Cristina Tîrnăucă

Dept. Matesco, Universidad de Cantabria

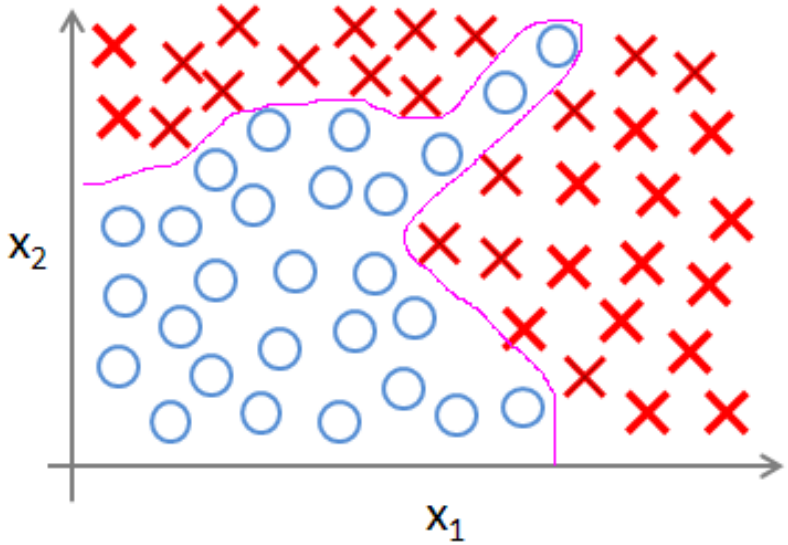
Fac. Ciencias – Grado en Ing. Informática

¿Para qué sirven?

Clasificación no lineal

¿Para qué sirven?

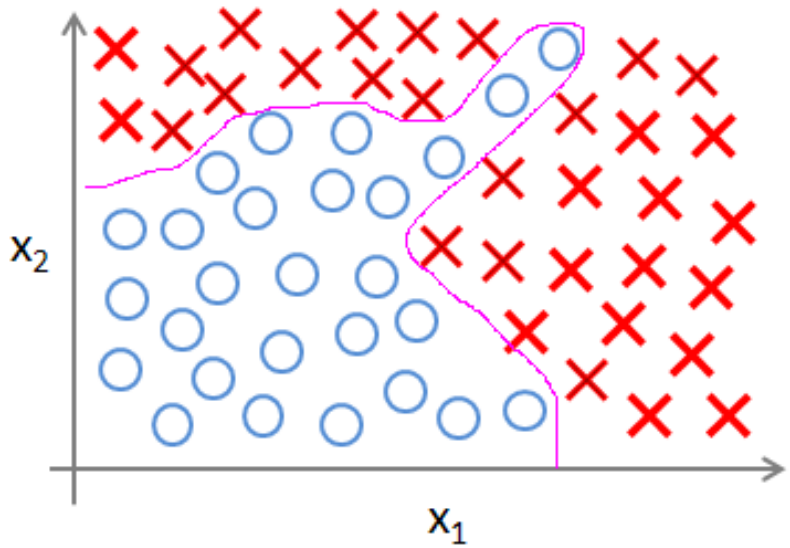
Clasificación no lineal



$$g(\theta_0 + \theta_1 x_1 + \theta_1 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1 x_2^2 + \theta_6 x_2^2 + \ldots)$$

¿Para qué sirven?

Clasificación no lineal



$$g(\theta_0 + \theta_1 x_1 + \theta_1 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1 x_2^2 + \theta_6 x_2^2 + \ldots)$$

¿Y si tenemos 100 atributos?

Redes neuronales

Origen: algoritmos que tratan de imitar el cerebro

- Muy populares en los años 80 y la primera parte de los 90
- Resurgimiento reciente: técnica state-of-the art para muchas aplicaciones

Redes neuronales

Origen: algoritmos que tratan de imitar el cerebro

- Muy populares en los años 80 y la primera parte de los 90
- Resurgimiento reciente: técnica state-of-the art para muchas aplicaciones

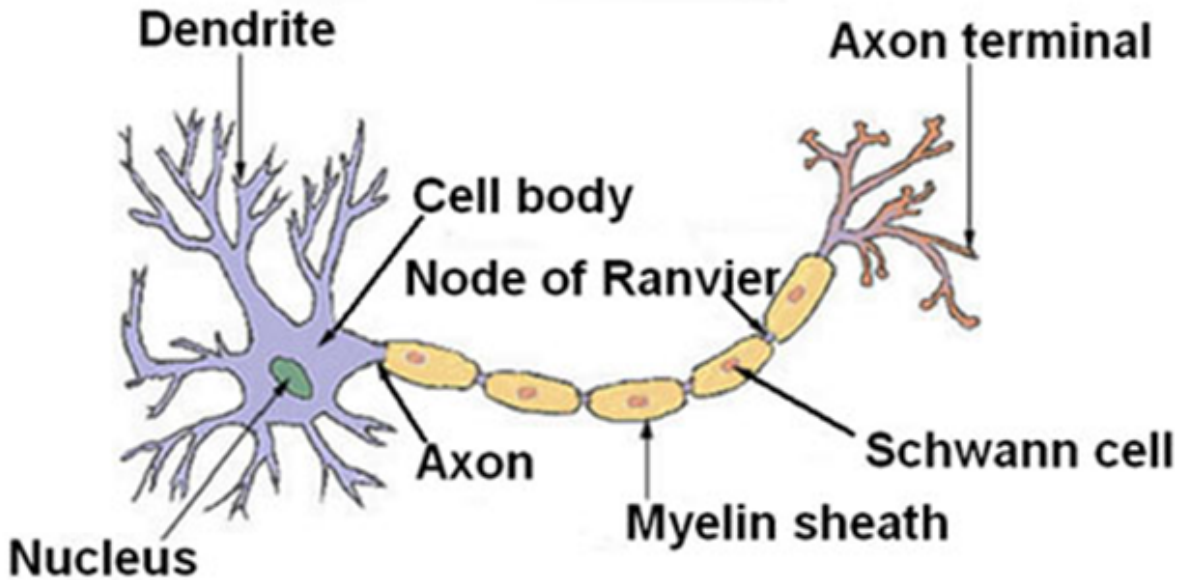
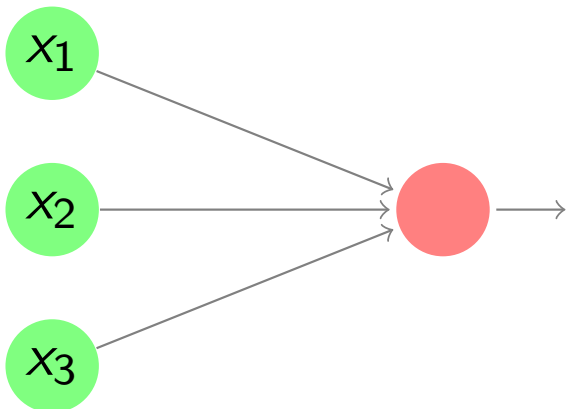


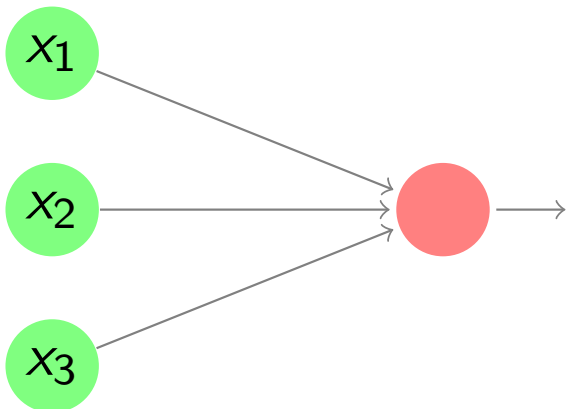
Figure: Structure of a typical neuron (Wikipedia, ©Quasar Jarosz)

El modelo neuronal: la unidad logística



$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

El modelo neuronal: la unidad logística

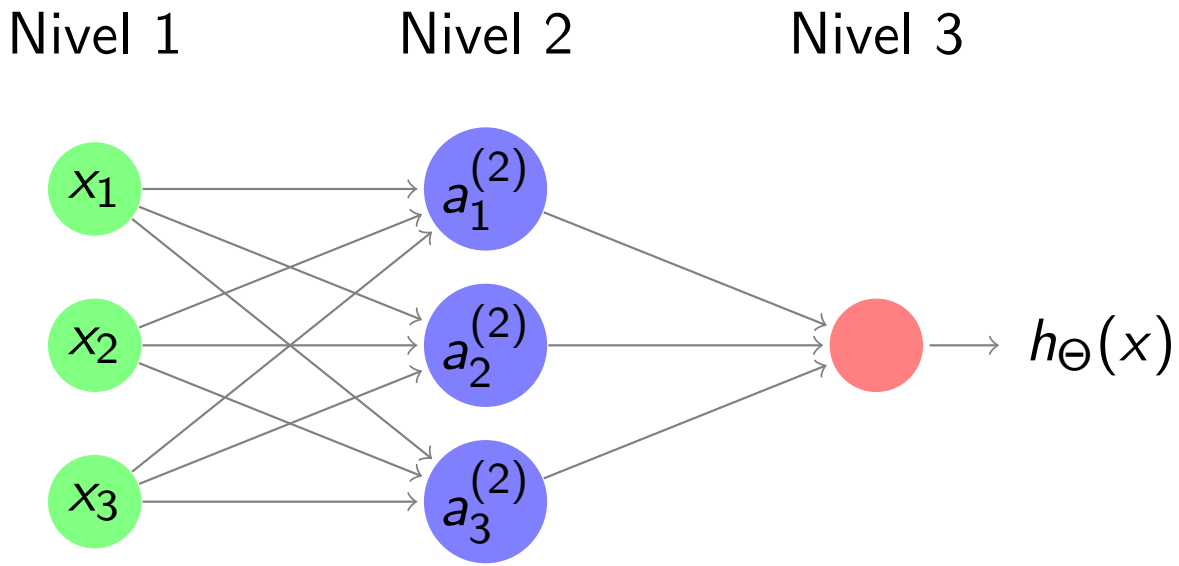


$h_\theta(x)$

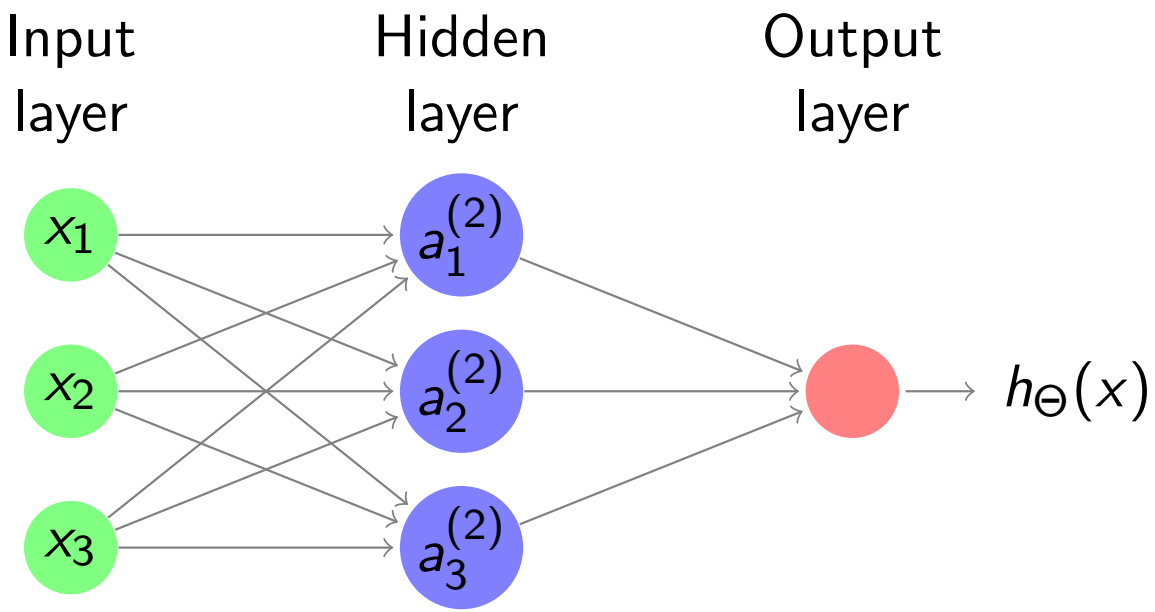
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

Función de activación logística: $h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$.

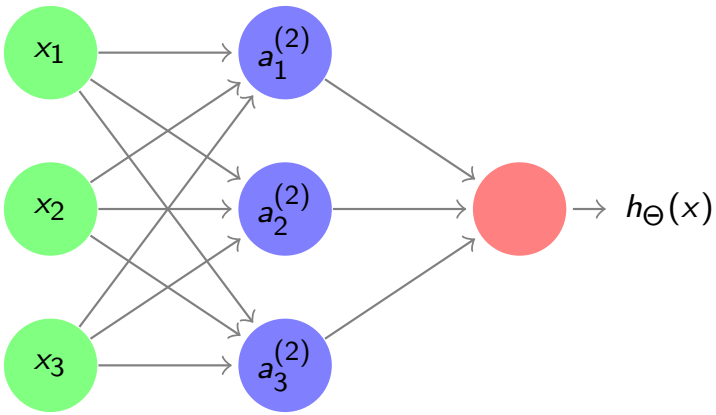
Red neuronal



Red neuronal



Red neuronal



$a_i^{(j)}$ = la activación de la unidad i en el nivel j

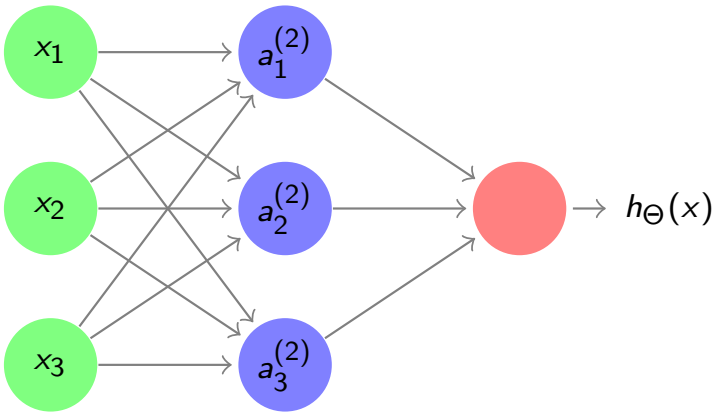
$\Theta^{(j)}$ = una matriz de pesos que controlan las transformaciones entre el nivel j y el nivel $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{12}^{(1)} x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{22}^{(1)} x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{32}^{(1)} x_3)$$

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

Si la red neuronal tiene s_j unidades en el nivel j, s_{j+1} unidades en el nivel $j+1$, la matriz $\Theta^{(j)}$ tendrá dimensión

Red neuronal



$a_i^{(j)}$ = la activación de la unidad i en el nivel j

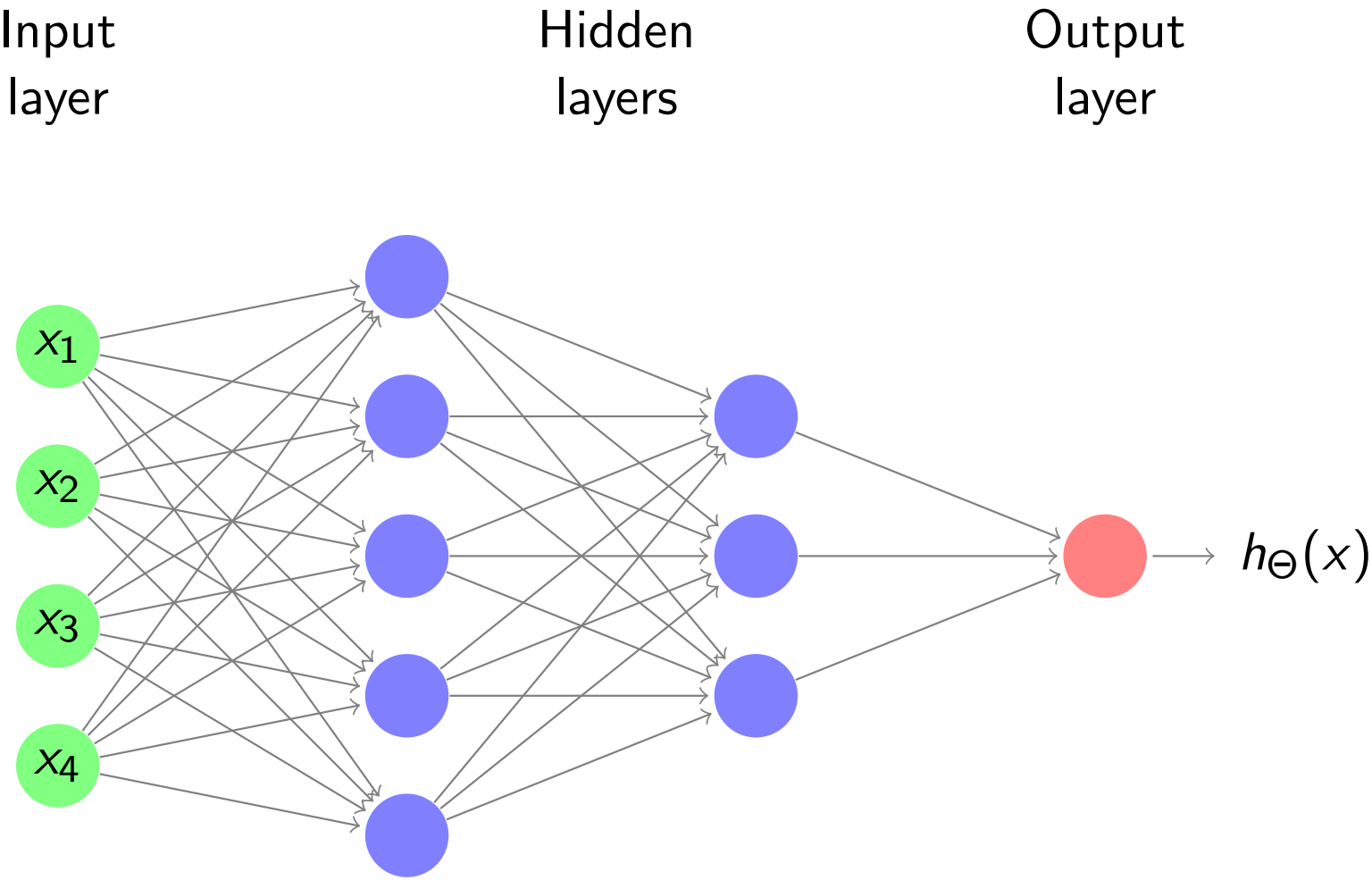
$\Theta^{(j)}$ = una matriz de pesos que controlan las transformaciones entre el nivel j y el nivel $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{12}^{(1)} x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{22}^{(1)} x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{32}^{(1)} x_3)$$

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

Si la red neuronal tiene s_j unidades en el nivel j, s_{j+1} unidades en el nivel $j+1$, la matriz $\Theta^{(j)}$ tendrá dimensión $s_{j+1} \times (s_j + 1)$.

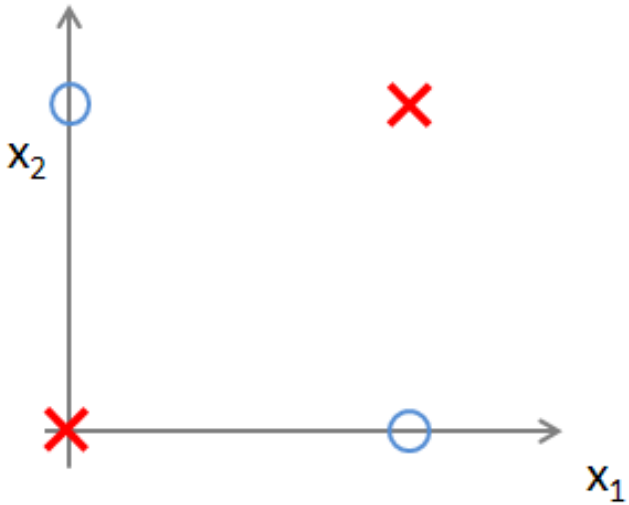
Otras arquitecturas de redes neuronales



Ejemplo de uso

Una red neuronal para XNOR

Supongamos que x_1, x_2 son binarios (0 o 1).



$y = x_1$ XNOR x_2
$y =$ NOT (x_1 XOR x_2)

Ejemplos de uso

Puerta AND

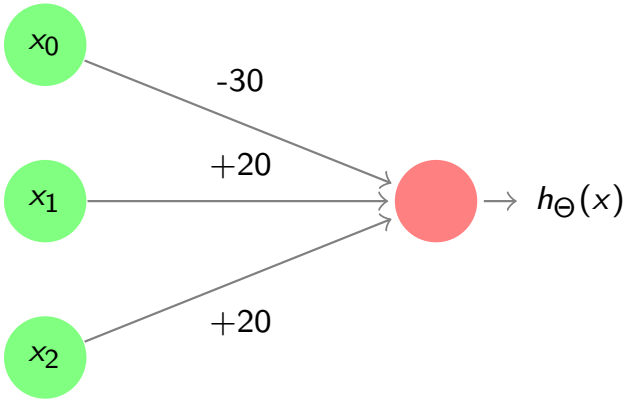
Puerta NOT

Puerta OR

(NOT x_1) AND (NOT x_2)

Ejemplos de uso

Puerta AND



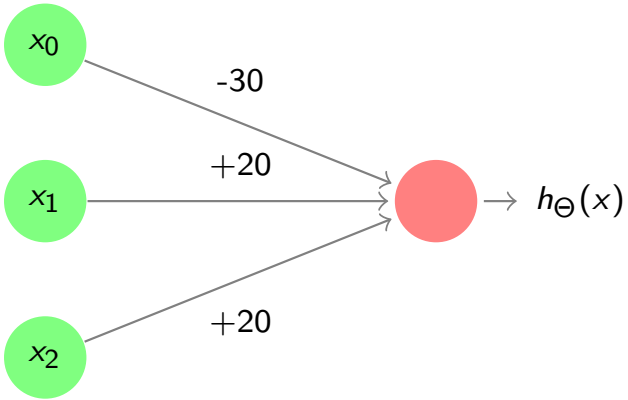
Puerta NOT

Puerta OR

(NOT x_1) AND (NOT x_2)

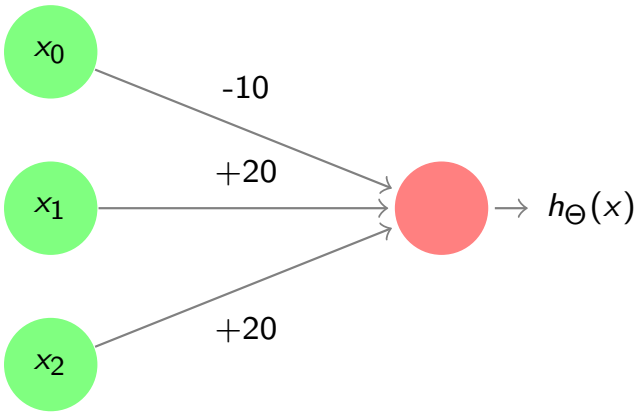
Ejemplos de uso

Puerta AND



Puerta NOT

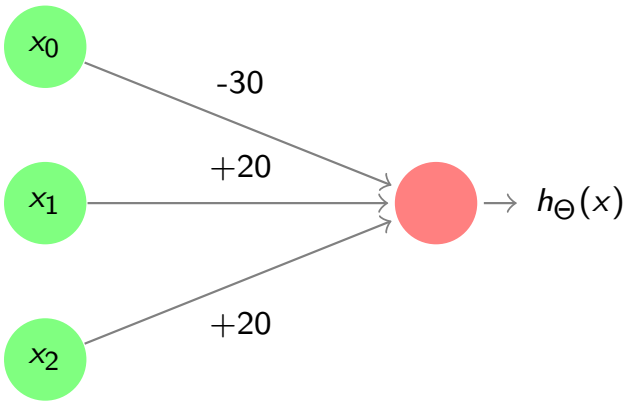
Puerta OR



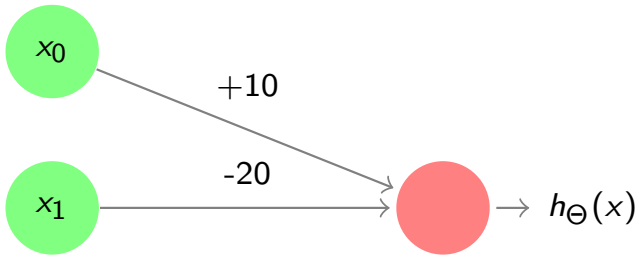
(NOT x_1) AND (NOT x_2)

Ejemplos de uso

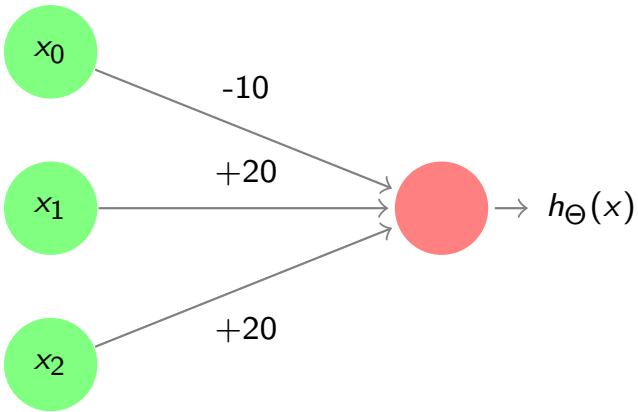
Puerta AND



Puerta NOT



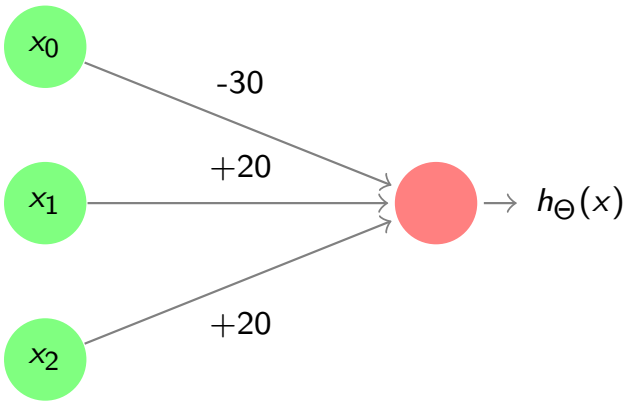
Puerta OR



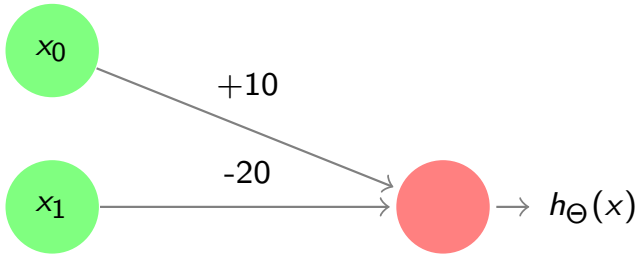
(NOT x_1) AND (NOT x_2)

Ejemplos de uso

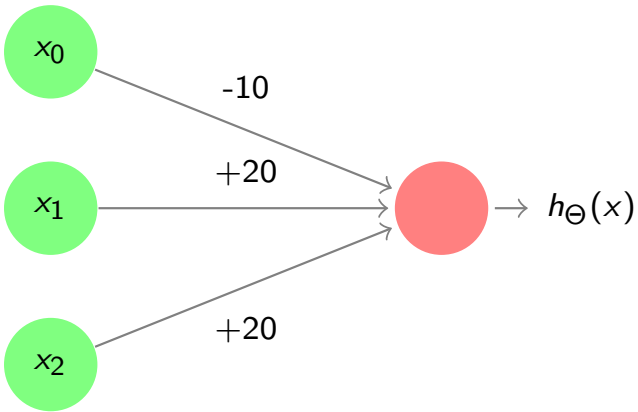
Puerta AND



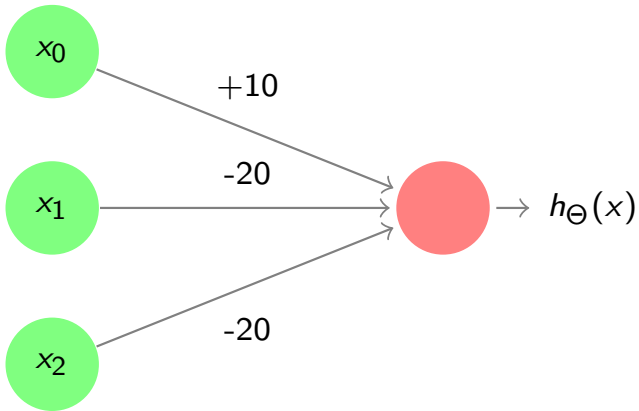
Puerta NOT



Puerta OR

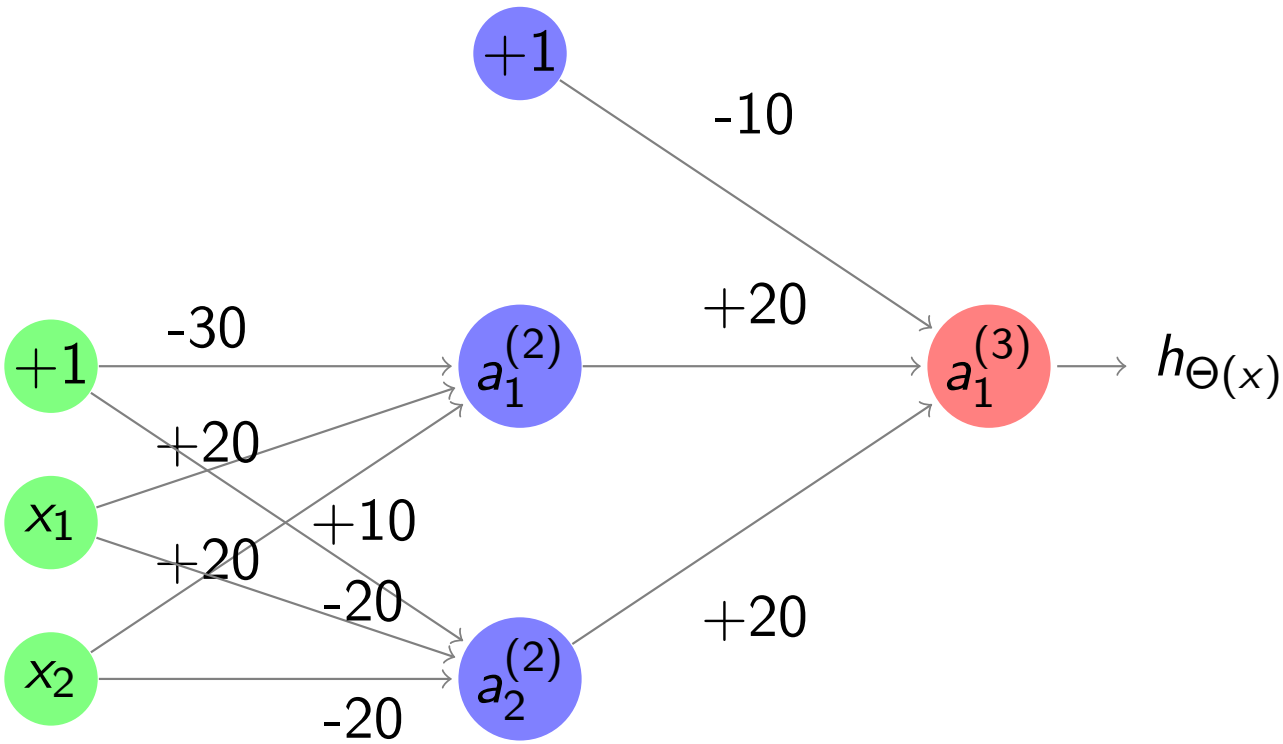


(NOT x_1) AND (NOT x_2)



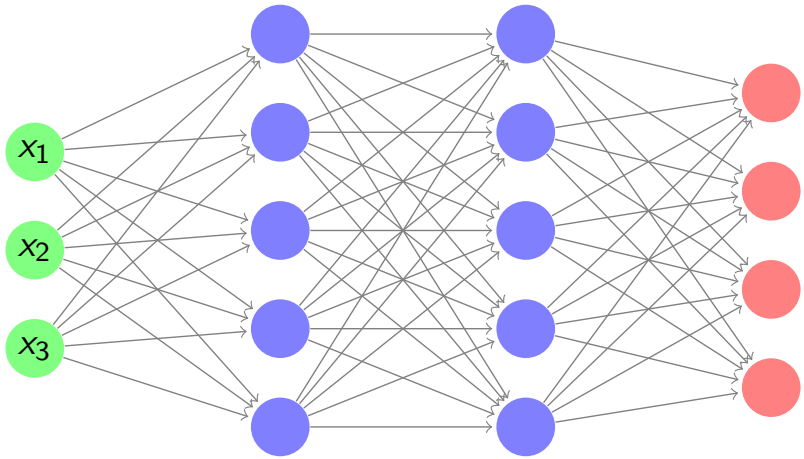
Una red neuronal para XNOR

Puerta XNOR



Redes neuronales - repaso

Clasificación no lineal



$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$

L = el número total de niveles en la red

s_l = el número total de unidades en el nivel l

Clasificación binaria	Clasificación multiclase
$y = 0$ o 1	$y \in \mathbb{R}^K$ (por ej., $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$)
1 unidad de salida	K unidades de salida

Función de costo

Regresión logística:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} * \log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)}) * \log\left(1 - h_\theta(x^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{p}\theta_j^2$$

Red neuronal:

$$h_\Theta(x) \in \mathbb{R}^K \qquad (h_\Theta(x))_i = \text{ la } i\text{-ésima salida}$$

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} * \log\left(h_\Theta(x^{(i)})_k\right) + (1 - y_k^{(i)}) * \log\left(1 - (h_\Theta(x^{(i)}))_k\right)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}} (\Theta_{ji}^{(l)})^2$$

Objetivo: encontrar Θ para que $J(\Theta)$ sea mínimo

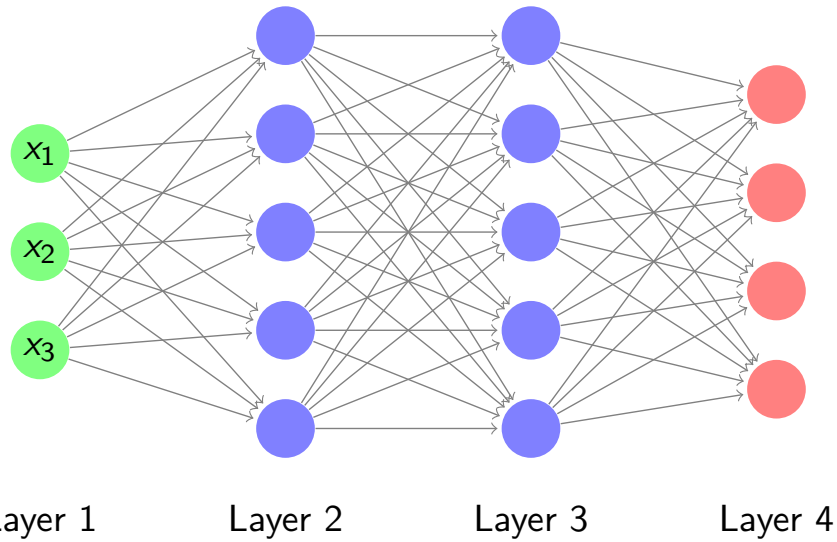
El método del gradiente descendente: hay que calcular $\frac{\partial}{\partial \Theta_{ji}^{(l)}} J(\Theta)$

Ejemplo

Como se calcula el gradiente para un ejemplo (x, y)

Propagación hacia adelante

$a^{(1)} = x$ (añadir x_0)
$z^{(2)} = \Theta^{(1)} a^{(1)}$
$a^{(2)} = g(z^{(2)})$ (añadir $a_0^{(2)}$)
$z^{(3)} = \Theta^{(2)} a^{(2)}$
$a^{(3)} = g(z^{(3)})$ (añadir $a_0^{(3)}$)
$z^{(4)} = \Theta^{(3)} a^{(3)}$
$a^{(4)} = g(z^{(4)})$
$h_\Theta(x) = a^{(4)}$



Propagación hacia atrás de los errores

$\delta_j^{(l)}$ = error del nodo j en el nivel l
$\delta^{(4)} = a^{(4)} - y$
$\delta^{(3)} = (\Theta^{(3)})^T \delta^{(4)} .* g'(z^{(3)})$, donde $g'(z^{(3)}) = a^{(3)} .* (1 - a^{(3)})$
$\delta^{(2)} = (\Theta^{(2)})^T \delta^{(3)} .* g'(z^{(2)})$, donde $g'(z^{(2)}) = a^{(2)} .* (1 - a^{(2)})$

Para $\lambda = 0$, $\frac{\partial}{\partial \Theta_{ji}^{(l)}} J(\Theta) = a_i^{(l)} \delta_j^{(l+1)}$

Propagación hacia atrás

(Backpropagation algorithm)

Conjunto de entrenamiento:
$\{(x^{(1)}, y^{(1)}, (x^{(2)}, y^{(2)}, \ldots, (x^{(m)}, y^{(m)}\}$

$\Delta_{ji}^{(l)} = 0$ (para todo l,j,i)
Para k de 1 a m
  $a^{(1)} = x^{(k)}$
  Calcula $a^{(l)}$ para $l = 2, 3, \ldots, L$
  Calcula $\delta^{(L)} = a^{(L)} - y^{(k)}$
  Calcula $\delta^{(l)}$ para $l = L-1, L-2, \ldots, 2$
  $\Delta_{ji}^{(l)} = \Delta_{ji}^{(l)} + a_i^{(l)} \delta_j^{(l+1)}$ (para todo l,j,i)
Para l de 1 a $L-1$
  Para j de 1 a s_{l+1}
    Para i de 0 a s_l

$$D_{ji}^{(l)} = \begin{cases} \frac{1}{m}\Delta_{ji}^{(l)} + \lambda\Theta_{ji}^{(l)} & \text{si } i > 0 \\ \frac{1}{m}\Delta_{ji}^{(l)} & \text{si } i = 0 \end{cases} \qquad \left(\frac{\partial}{\partial \Theta_{ji}^{(l)}} J(\Theta) = D_{ji}^{(l)} \right)$$

Aproximación numérica del gradiente

$$\frac{\partial}{\partial \Theta_1} J(\Theta) = \frac{J(\Theta_1+\varepsilon,\Theta_2,\ldots,\Theta_n)-J(\Theta_1-\varepsilon,\Theta_2,\ldots,\Theta_n)}{2\varepsilon}$$

$$\ldots$$

$$\frac{\partial}{\partial \Theta_n} J(\Theta) = \frac{J(\Theta_1,\Theta_2,\ldots,\Theta_n+\varepsilon)-J(\Theta_1,\Theta_2,\ldots,\Theta_n-\varepsilon)}{2\varepsilon}$$